(11) Japanese Patent Laid-Open No. 09-098415

(43) Laid-Open Date: April 8, 1997

(21) Application Number: 07-252485

(22) Filing Date: September 29, 1995

(71) Applicant: IKEGAMI TSUSHINKI CO LTD

(72) Inventor: HIDETAKA IMAI

(72) Inventor: HODAKA MIZUGUCHI

(72) Inventor: YUJI HARADA

---

(54) [Title of the Invention]

METHOD AND DEVICE FOR PERFORMING QUANTIZATION AND INVERSE QUANTIZATION

(57) [Abstract]

[Problem to be Solved]

To enable quantization/inverse quantization using a simple construction.

[Solution]

A quantized data occupancy of a buffer memory 14 is compared with a target occupancy using a occupancy comparator 3, and a difference is accumulated using an accumulator 5. Correction information within a predetermined range is generated using a coefficient multiplier 7 from an accumulation result and fed back to a code amount comparator 6A to allow correction to keep a target amount of code within a predetermined range. With this arrangement, the target amount of

code can be set to an integer value and a hardware construction is simplified.

[Claims for the Patent]

[Claim 1]

A method for performing quantization characterized by:

when transmitting quantized data obtained by quantizing inputted component data,

accumulating an error, with respect to a predetermined target amount of code, in an amount of generated code included in a block that is a unit corresponding to a plurality of the component data;

controlling the amount of generated code using an accumulation value of error with respect to the target amount of code;

accumulating an error, with respect to a predetermined target occupancy, in an occupancy by the quantized data of a buffer memory that stores quantized data after the control; and

correcting the target amount of code using the accumulation value of error with respect to target occupancy.

[Claim 2]

The method for performing quantization according to claim 1, characterized by:

when performing the control,

calculating a scale factor by executing predetermined calculation processing on the accumulation value of error with respect to the target amount of code;

setting the divisors for quantization by multiplying by the scale factor quantization table values preset for each block; and

transmitting the accumulation value of error with respect to the target amount of code to an inverse quantization device side only every N frames (where N is an integer greater than or equal to 1).

[Claim 3]

The method for performing quantization according to claim 2, characterized by

transmitting the accumulation value of error with respect to the target occupancy to the inverse quantization device side only every N frames.

[Claim 4]

The method for performing quantization according to claim 2, characterized by

when calculating the amount of generated code,

calculating a number of encoded bits based on pair data made up of zero run-lengths and non-zero data representing M pieces of data (where M is an integer greater than or equal to 1) in a single block.

[Claim 5]

A method for performing inverse quantization by

multiplying quantized code data transmitted from a quantization device side by a predetermined multiplier, characterized by:

accumulating an error, with respect to a target amount of code used on the quantization device side, in an amount of generated code for each block corresponding to a plurality of component data based on the quantized code data from a buffer memory which stores the quantized code transmitted from the quantization device side;

generating a multiplier using the accumulation value of error with respect to the target amount of code;

accumulating an error, with respect to a predetermined target occupancy, in an occupancy by the quantized data in said buffer memory; and

correcting the target amount of code using the accumulation value of error with respect to the target occupancy.

[Claim 6]

The method for performing inverse quantization according to claim 5, further characterized by:

when accumulating the error with respect to the target amount of code,

setting the accumulation value of error with respect to the target amount of code so as to equal the accumulation value transmitted from the quantization

device side every N frames (when N is an integer greater than or equal to 1);

calculating a scale factor by implementing calculation processing on the accumulation value in a same manner as on the quantization device side; and

setting the multiplier for inverse quantization by multiplying inverse quantization table values, which have same values as in a quantization table used on the quantization device side, by the scale factor.

[Claim 7]

The method for performing inverse quantization according to claim 6, characterized by

when calculating the amount of generated code,

calculating a number of encoded bits based on pair data made up of zero run-length data and non-zero data obtained by zero run-length decoding.

[Claim 8]

A quantization device for transmitting quantized data obtained by quantizing inputted component data, characterized by having:

first accumulation means for obtaining an accumulation value by accumulating an error, with respect to a predetermined target amount of code, in an amount of generated code included in a block that is a unit corresponding to a plurality of the component data;

control means for controlling the amount of

generated code using the accumulation value of error with respect to the target amount of code;

a buffer memory for storing quantized data after the control;

second accumulation means for obtaining an accumulation value by accumulating an error, with respect to a predetermined target occupancy, in an occupancy of the buffer memory by the quantized data; and

correction means for correcting the target amount of code using the accumulation value of error with respect to the target occupancy.

[Claim 9]

The quantization device according to claim 8, characterized by said control means having:

scale factor calculating means for calculating a scale factor by executing predetermined calculation processing on the accumulation value of error with respect to the target amount of code;

divisor calculating means for calculating the divisors for quantization by multiplying by the scale factor quantization table values preset for each block; and

accumulation value transmitting means for transmitting the accumulation value to an inverse quantization device side only every N frames (where N is an integer greater than or equal to 1).

[Claim 10]

The quantization device according to claim 9, characterized by the amount of generated code being an number of encoded bits calculated based on pair data made up of zero run-lengths and non-zero data representing M pieces of data (where M is an integer greater than or equal to 1) in a single block.

[Claim 11]

An inverse quantization device for performing inverse quantization by multiplying quantized code data transmitted from a quantization device side by a predetermined multiplier, the inverse quantization device characterized by having:

a buffer memory for storing the quantized code data transmitted from the quantization device side;

first accumulation means for obtaining an accumulation value by accumulating error, with respect to a target amount of code used on the quantization device side, in an amount of generated code for each block corresponding to a plurality of component data based on the quantized code data from said buffer memory;

generation means for generating a multiplier using the accumulation value of error with respect to the target amount of code;

second accumulation means for obtaining an accumulation value by accumulating error, with respect

to a predetermined target occupancy, in an occupancy of the buffer memory by the quantized data; and

correction means for correcting the target amount of code using the accumulation value of error with respect to the target occupancy.

[Claim 12]

The inverse quantization device according to claim 11, characterized by said multiplier generation means having:

accumulation means including an accumulation circuit for accumulating an error with respect to the target amount of code, and an accumulation value setting circuit for setting an accumulation value of said accumulation circuit by introducing an accumulation value transmitted from the quantization device side every N frames (where N is an integer greater than or equal to 1);

scale factor calculating means for calculating a scale factor by implementing calculation processing on the accumulation value obtained from said accumulation means in a same manner as on the quantization device side; and

multiplier calculation means for calculating the multipliers for inverse quantization by multiplying by the scale factor inverse quantization table values, which have same values as in a quantization table used on the quantization device side.

[Claim 13]

The inverse quantization device according to claim 12, characterized by

the amount of generated code being an number of encoded bits calculated based on pair data made up of zero run-lengths and non-zero data obtained by zero run-length decoding.

[Detailed Description of the Invention]

[0001]

[Field of the Invention]

The present invention relates to a method and device for performing favorable quantization/inverse quantization for compressed transmission of image data or the like.

[0002]

[Conventional Art]

First, generally known techniques for quantization and inverse quantization are described.

[0003]

Quantization refers to the process of dividing component output values by component-specific numerical values and rounding off at the decimal point. Because of the rounding off, the original component values are not completely recovered when the quantized values are multiplied by the numerical values at inverse quantization. The image quality and compression ratio

for compressed images is determined by the quantization processing. For a high image quality, the component output values should be divided by small values. However, this gives a low compression ratio. Conversely, since dividing by a large value gives a smaller effective number of bits, the compression ratio increases at the cost of a lower image quality. In quantization, if a constant numerical value is used in the division, the resulting amount of code varies according to the inputted image.

[0004]

To avoid this problem, a technique in which the values used in the division is varied depending on the inputted image to maintain a constant amount of outputted code is conceivable. However, in the quantization and inverse quantization, the same numerical value must be used in the division and the multiplication. Therefore, the value (scale factor) used in the quantizer must be transmitted to the inverse quantizer side.

[0005]

The following describes a conventional technique using an example in which an amount of code in image encoding is controlled.

[0006]

For the simplest possible illustration of this technique, a plurality of quantizers 110A to 110C, a

plurality of variable length encoders 111A to 111C and a buffer memory 112 are provided, as shown in Figure 10 for instance. An output selector 114 is controlled to select, using a judgment unit 113, the quantizer with a suitable amount of code, according to an accumulation value of error with respect to a target value for occupancy of the buffer memory 112.

[0007]

In an alternative method, as shown in Figure 11, a code amount predicting unit 120 uses some method to predict the amount of code (such as spatial information or frequency information) in the image data, a quantizer 122 is controlled on the basis of the prediction and occupancy of a buffer memory 121, and quantized data is transmitted after control in a variable length encoder 123.

[0008]

The above-described scale factor is applied to an entire quantization table, and the amount of information on the transmission path is controlled to be kept constant by appropriate changes in the value of the scale factor.

[0009]

The scale factor must be transmitted on the transmission path, since if the decoder is unaware of the scale factors used by the encoder, the numerical value used in the division will be unknown, and

decoding will be impossible. When a scale factor is not used by the encoder (the case in which the scale factor is constant), the amount of information on the transmission path depends on the input data, making it impossible to maintain a constant amount of information on the transmission path.

[0010]

MPEG2 is an international standard format used for the compression of image data. According to this standard (ISO/IEC 13818-2 "Information Technology - Generic coding of moving pictures and associated audio information - Draft International Standard 1994) the scale factor can be changed for each macroblock. In Y:Cb:Cr = 4:2:2 coding, 5 bits are allocated for the scale factor of a macroblock that includes a total of 8 blocks made up of 4 blocks in Y, 2 blocks in Cb, and 2 blocks in Cr.

[0011]

Even when all data but the scale factors in MPEG 2 are ignored, for a high vision signal it is necessary to transmit 1920 (horizontal number of pixels) × 1036 (vertical number of pixels) × 5 (bit) × 30 (frames/sec) × 2 (4:2:2 format)/64 (number of pixels in one block)/8 (blocks) = 1165500 (bit/sec), which is an enormous amount of data.

[0012]

[Problems to be Solved by the Invention]

As described above, one problem with the conventional technology of the construction shown in Figure 10 is that to improve accuracy of the control a large number of quantizers have to be provided, and so the scale of the hardware increases. When the construction of Figure 11 is used in an attempt to solve this problem, a complicated mechanism is required to predict the amount of code, and, as might be expected, no reduction in the scale of the hardware is achieved.

[0013]

A further problem is that when transmitting scale factors conventionally, a large amount of scale factor information must be transmitted on a transmission path of limited capacity.

[0014]

Moreover, as the detail of the image to be transmitted increases, the data capacity required for transmission of the scale factors becomes extremely large.

[0015]

To solve these problems, it is the object of the present invention to provide method and device which allow quantization/inverse quantization using a simple construction which requires neither the plurality of quantizers nor the special mechanism for predicting the amount of code. It is a further object to provide a

method and a device for performing quantization/inverse quantization which realize an improvement in the transmission efficiency of required data by sharply reducing, in comparison with conventional techniques, the data capacity required to transmit the scale factors.

[0016]

[Means for Solving the Problems]

To solve these problems, the invention according to claim 1 is characterized by, when transmitting quantized data obtained by quantizing inputted component data, accumulating an error, with respect to a predetermined target amount of code, in an amount of generated code included in a block that is a unit corresponding to a plurality of the component data; controlling the amount of generated code using an accumulation value of error with respect to the target amount of code; accumulating an error, with respect to a predetermined target occupancy, in an occupancy by the quantized data of a buffer memory that stores quantized data after the control; and correcting the target amount of code using the accumulation value of error with respect to target occupancy.

[0017]

Further the invention according to claim 2 is the invention according to claim 1 further characterized by, when performing the control, calculating a scale factor

by executing predetermined calculation processing on the accumulation value of error with respect to the target amount of code; setting the divisors for quantization by multiplying, by the scale factor, quantization table values preset for each block; and transmitting the accumulation value of error with respect to the target amount of code to an inverse quantization device side only every N frames (where N is an integer greater than or equal to 1).

[0018]

The invention according to claim 3 is the invention according to claim 2 further characterized by transmitting the accumulation value of error with respect to the target occupancy to the inverse quantization device side only every N frames.

[0019]

The invention according to claim 4 is the invention according to claim 2 further characterized by, when calculating the amount of generated code, calculating a number of encoded bits based on pair data made up of zero run-lengths and non-zero data representing M pieces of data (where M is an integer greater than or equal to 1) in a single block.

[0020]

The invention of claim 5 is, for performing inverse quantization by multiplying quantized code data transmitted from a quantization device side by a

predetermined multiplier, characterized by: accumulating an error, with respect to a target amount of code used on the quantization device side, in an amount of generated code for each block corresponding to a plurality of component data based on the quantized code data from a buffer memory which stores the quantized code transmitted from the quantization device side; generating a multiplier using the accumulation value of error with respect to the target amount of code; accumulating an error, with respect to a predetermined target occupancy, in an occupancy by the quantized data in said buffer memory; and correcting the target amount of code using the accumulation value of error with respect to the target occupancy.

[0021]

The invention according to claim 6 is the invention according to claim 5 further characterized by: when accumulating the error with respect to the target amount of code, setting the accumulation value of error with respect to the target amount of code so as to equal the accumulation value transmitted from the quantization device side every N frames (when N is an integer greater than or equal to 1); calculating a scale factor by implementing calculation processing on the accumulation value in a same manner as on the quantization device side; and setting the multiplier for inverse quantization by multiplying inverse

quantization table values, which have same values as in a quantization table used on the quantization device side, by the scale factor.

[0022]

The invention according to claim 7 is the invention according to claim 6 further characterized by, when calculating the amount of generated code, calculating a number of encoded bits based on pair data made up of zero run-length data and non-zero data obtained by zero run-length decoding.

[0023]

The present invention according to claim 8 is a quantization device for transmitting quantized data obtained by quantizing inputted component data, characterized by having: first accumulation means for obtaining an accumulation value by accumulating an error, with respect to a predetermined target amount of code, in an amount of generated code included in a block that is a unit corresponding to a plurality of the component data; control means for controlling the amount of generated code using the accumulation value of error with respect to the target amount of code; a buffer memory for storing quantized data after the control; second accumulation means for obtaining an accumulation value by accumulating an error, with respect to a predetermined target occupancy, in an occupancy of the buffer memory by the quantized data;

and correction means for correcting the target amount of code using the accumulation value of error with respect to the target occupancy.

[0024]

The invention according to claim 9 is the invention according to claim 8 further characterized by said control means having: scale factor calculating means for calculating a scale factor by executing predetermined calculation processing on the accumulation value of error with respect to the target amount of code; divisor calculating means for calculating the divisors for quantization by multiplying by the scale factor quantization table values preset for each block; and accumulation value transmitting means for transmitting the accumulation value to an inverse quantization device side only every N frames (where N is an integer greater than or equal to 1).

[0025]

The invention according to claim 10 is the invention according to claim 9 further characterized by the amount of generated code being an number of encoded bits calculated based on pair data made up of zero run-lengths and non-zero data representing M pieces of data (where M is an integer greater than or equal to 1) in a single block.

[0026]

The invention according to claim 11 is an inverse quantization device for performing inverse quantization by multiplying quantized code data transmitted from a quantization device side by a predetermined multiplier, the inverse quantization device characterized by having: a buffer memory for storing the quantized code data transmitted from the quantization device side; first accumulation means for obtaining an accumulation value by accumulating error, with respect to a target amount of code used on the quantization device side, in an amount of generated code for each block corresponding to a plurality of component data based on the quantized code data from said buffer memory; generation means for generating a multiplier using the accumulation value of error with respect to the target amount of code; second accumulation means for obtaining an accumulation value by accumulating error, with respect to a predetermined target occupancy, in an occupancy of the buffer memory by the quantized data; and correction means for correcting the target amount of code using the accumulation value of error with respect to the target occupancy.

[0027]

The invention according to claim 12 is the invention according to claim 11 further characterized by the multiplier generation means having: accumulation means including an accumulation circuit for

accumulating an error with respect to the target amount of code, and an accumulation value setting circuit for setting an accumulation value of the accumulation circuit by introducing an accumulation value transmitted from the quantization device side every N frames (where N is an integer greater than or equal to 1); scale factor calculating means for calculating a scale factor by implementing calculation processing on the accumulation value obtained from said accumulation means in a same manner as on the quantization device side; and multiplier calculation means for calculating the multipliers for inverse quantization by multiplying by the scale factor inverse quantization table values, which have same values as in a quantization table used on the quantization device side.

[0028]

The invention according to claim 13 is the invention according to claim 12 further characterized by the amount of generated code being an number of encoded bits calculated based on pair data made up of zero run-lengths and non-zero data obtained by zero run-length decoding.

[0029]

[Embodiments of the Invention]

On an encoder (100 in Figure 1) side, a method is used to calculate an amount of code to be transmitted for a single block, accumulate a difference between the

calculation result and a target amount of code for the block, multiply the accumulation value by a predetermined coefficient, add a predetermined coefficient, and thereby determine the scale factor for quantization of the next block. The encoder further corrects the target amount of code so that an occupancy of data in a buffer memory storing the data to be transmitted becomes a target occupancy.

[0030]

On a decoder (200 in Figure 1) side, predetermined coefficients of the scale factor determining operation on the encoder side are held, a circuit is provided to perform similar operations to those of the encoder, an amount of code in the transmitted block is measured, and that amount is supplied to the circuit resembling the encoder side. The scale factor for inverse quantization of the next block is then determined based on an accumulation value for the continuously accumulated difference to a same target amount of code for the block as on the encoder side. Further, the target amount of code is corrected so that the data occupancy in the buffer memory storing the transmitted data reaches a predetermined amount.

[0031]

According to the above construction, quantization/inverse quantization is possible using a simple construction. Moreover, when the above

construction is used and both initial accumulation values are set to the same value, subsequent operations will be identical and the same scale factor will be obtained for corresponding blocks. In other words, when the accumulation values are the same, the same scale factor values are obtained. In subsequent blocks too, if the amount of code for the blocks is the same, the accumulation values will also be the same.

[0032]

Hence, fundamentally, matching of subsequent scale factors can be performed by simply initializing the accumulation values in the encoder and the decoder at the start of the processing. However, the encoder and decoder are generally located in different places and will not be switched on simultaneously. Therefore, to ensure that both accumulation values agree at any instant despite the lack of connection, it is necessary to regular perform processing to make the two agree. In the present embodiment the accumulation value is transmitted every four frames, with four frames forming a processing unit. Naturally, between transmissions, the scale factors for each block in the frames are made to agree by way of the amounts of code in the blocks. It is therefore possible to freely vary the scale factor for each block unit, and thereby finely control the amount of code in accordance with the image. Also, since this processing is performed on block units,

where a single frame is constructed from a large number of blocks, it is possible to set a comparatively suitable amount of code without having to perform an advance check on the image of the entire frame.

[0033]

The following is a more detailed explanation of the above described embodiment.

[0034]

1. To determine the scale factor, the differences between the target amount of code and a calculated amount of code in the block are accumulated for each block. The accumulation value is multiplied by a predetermined coefficient, a predetermined coefficient is added to the value, and the scale factor to be used when quantizing the next block is outputted. The construction for this processing is the same in the encoder and the decoder, and the target amount of code and the predetermined coefficients are also set to be the same in the encoder and the decoder. More specifically, the encoder has a function for outputting the value from the accumulator, and the decoder has a function for receiving input of the accumulation value from the encoder and setting a value in an accumulator.

[0035]

2. On the decoder side, amounts of code for each block in the transmitted data stream are totaled. The resulting total amount of code is then supplied to the

above described scale factor determining circuit.

[0036]

3. On the decoder side, the accumulation value transmitted from the encoder is received and supplied to the scale factor determining circuit. The accumulator of the scale factor determining circuit is then set to the received accumulation value. With this arrangement, the scale factor for the next block for which the amount of code has been totaled in the decoder, is the same value as in the encoder.

[0037]

4. The difference between the amount of code in a block and the target amount of code is calculated for a each block. The accumulation result for each block is then multiplied by the predetermined coefficient and the predetermined coefficient is added to the result to determine the scale factor. If the accumulation value, the coefficient used in the multiplication, and the coefficient used in the addition are the same, the obtained scale factor will also be the same.

[0038]

At the beginning of the data for a frame, the accumulation value data resulting from the accumulation of the amount of code in the last block of the previous frame is transmitted. On the encoder side, processing for the next frame is performed with the accumulation value data as an initial value. On the decoder side,

the processing of the directly following frame is processed with the received same data as an initial value for the accumulation value data. In the processing of frames thereafter, once the processing for a given block has been completed, the calculated amount of code for the given block is used to update the accumulation value, and the new accumulation value gives the scale factor for quantization of the next block.

[0039]

In the decoder, a given block is received, a code length of pair data is obtained in the same way as in the encoder from the pair data of Huffman decoded run-length code, and the code length is calculated by summing the obtained code lengths for the block unit. The result of the calculation is then transmitted to scale factor determining circuit, at which point the accumulation value is determined. In other words, the scale factor for the inverse quantization of the next block is determined.

[0040]

When a plurality of blocks has been transmitted and the transmission a frame has been completed, the processing for the next frame is started. Every four frames, the accumulation value is transmitted from the encoder side to the decoder side. Setting the encoder side to the received accumulation value allows

quantization of the subsequent first block using a scale factor based on the received accumulation value.

[0041]

5. On the encoder side, an error, with respect to a target occupancy, of an occupancy by data of a buffer memory for storing the quantized data to be transmitted is calculated. By using the accumulation value to correct the target amount of code every four frames or the like, the data occupancy in the buffer memory is corrected to the target amount. The same process is performed on the decoder side.

[0042]

When the encoder and the decoder are operated using the above described method, if the accumulation values are the same at the beginning of a frame then the quantization and inverse quantization for subsequent blocks will be performed using the same scale factor. Fundamentally, if the accumulation values are made to match at the beginning, as long as there is no damage to the transmission path, the transmitted data will be accurately received by the decoder, and inverse quantization will be performed in the decoder using the scale factor that was used in the encoder.

[0043]

When the condition of the transmission path is poor and a transmission error or the like occurs, a

difference will appear between the measurements of the amount of code in the encoder and the decoder. With the present invention, even if such conditions do occur, since periodic setting is performed by transmitting the accumulation value for the run-length code, the abnormal conditions are only allowed to last for a short time after which the scale factor will once again be made to match in the encoder and the decoder.

[0044]

[Embodiments]

The following describes a first embodiment of the present invention in detail with reference to the drawings.

[0045]

Figure 1 is a block diagram showing part of a quantizer to which the present invention is applied in an encoder and a decoder. Parts not related to the present invention, such as the video processing parts preceding the quantizer, have been omitted. Quantization itself is performed in the block corresponding to a divider 2. In the drawing, the frame 100 marked by the dashed line is the encoder. Of the blocks included in the encoder 100, reference numeral 2 denotes a divider which receives input of component data which is image data. A detailed circuit construction for the divider 2 is shown in Figure 5. Reference numeral 4 denotes a circuit to perform

coding/code amount calculation processing. A detailed construction for this circuit is shown in Figure 6. Reference character 6A denotes a code amount comparator. A detailed construction for this circuit is shown in Figure 2. Reference character 8A denotes an accumulator. A detailed construction for this circuit is shown in Figure 3A. Reference character 10A denotes a scale factor calculator. A detailed construction for this circuit is shown in Figure 4.

[0046]

Further, reference numeral 12 denotes a Huffman encoding circuit which receives input of coded data (pair data), and reference numeral 14 denotes a buffer memory for maintaining a constant rate of transmission.

[0047]

Reference numeral 3 denotes an occupancy comparator, and reference numeral 7 denotes a coefficient multiplier. Detailed constructions of these circuits are shown in Figure 9.

[0048]

Reference numeral 16-1 denotes a digital format transmission path which is connected to the encoder 100. The transmission path 16-1 may be a digital data recording and playback device such as a digital VTR. Reference numeral 16-2 (and reference numeral 16-3) denotes a data line of the present invention for matching an accumulation value of error with respect to

the target amount of code in the encoder and the decoder. Reference numeral 16-4 (and reference numeral 16-5) denotes a data line of the present invention for matching an accumulation value of error with respect to a target amount of code in the encoder and the decoder. These data are multiplexed with the encoded image data and transmitted on the transmission path 16-1.

[0049]

The frame 200 marked by the dashed line below the encoder 100 denotes the decoder. The reference character 6B in the decoder 200 denotes a code amount comparator which has the same construction as the code amount comparator 6A (see Figure 2) on the encoder side. Reference character 8B denotes an accumulator. A detailed construction for this circuit is shown in Figure 3B. Reference character 10B denotes a scale factor calculator which has the same construction as the scale factor calculator 10A (see Figure 4) on the encoder side. Reference numeral 18 denotes a buffer memory, and reference numeral 19 denotes a Huffman decoding circuit. Reference numeral 20 denotes a circuit which performs decoding/code amount calculation processing. A detailed construction for this circuit is shown in Figure 7. Reference numeral 22 denotes a multiplier which outputs decoded component data. A detailed construction for this circuit is shown in Figure 8.

[0050]

Reference numeral 11 denotes an occupancy comparator, reference number 13 denotes an accumulator, and reference numeral 15 denotes a coefficient multiplier. Detailed constructions of these circuits are shown in Figure 9.

[0051]

The following describes operations of the present embodiment in detail, with reference to Figure 1 and Figures 1 through 9.

[0052]

Firstly, the component data inputted to the divider 2 is data (12 bit) resulting from performing an orthogonal transform on the image data and resolving the result into frequency components. The data format makes use of blocks with 64 pieces of 2's complement, 12-bit data to a block. A high vision signal, for instance, has 1920 × 1036/64 = 31080 blocks per frame.

[0053]

In the divider 2, inputted component data is divided (using the scale factor × each element in a quantization table), and then undergoes rounding processing (see Figure 5). Since, after rounding, the component data cannot be returned to an original form in the decoder, image quality is affected. Dividing the component data by large numbers gives an increase in the number of zeros, causing the compression ratio

to be large and the image quality to drop.

[0054]

In encoding/code amount calculation processing circuit 4 of the next stage, run-length encoding (60Y, 60U, 60V) is performed, as shown in Figure 6, and the amount of code in single block is calculated (62Y, 62U, 62V, 64, and 66). In the run-length encoding, the zeros (zero run-lengths) in the division data are separated to form non-zero pair data. For the calculation of the amount of code in each block, a calculation of bit length is performed for each Huffman coded block by summing, at the pair data stage, bit lengths corresponding to the pair data in each block. The calculation of the amount of code is performed simultaneously with the Huffman coding.

[0055]

In the processing to calculate the amount of code, the total bit length for a given block is calculated by consulting the table 1 below. In table 1, the 1/2 pair indicates a zero run-length of 1, a non-zero run-length of 2, and has a generated bit-length of 6 bits. In addition, 2 additional bits indicating an absolute value of the non-zero data are transmitted. The data length of the component data is 12 bits. Therefore when the pair data 1/2 is generated, the data is compressed to (6 + 2)/(12 + 12) = 1/3 of an original length.

[0056]

[Table 1]

[0057]

In the code amount comparator 6A, a target amount of code calculated as difference (described later) with a correction information amount in an adder 25 is compared with an amount of generated code using the adder 26, as shown in Figure 2. The amount of generated code described here refers to the amount of data in a block after the block has been quantized using a scale factor determined using data from the accumulator 8A updated after processing the preceding block, run-length encoded, and Huffman encoded, and is expressed in bits. The target amount of code refers to a target amount of code per block and is expressed in bits. The result of subtracting the generated amount of code from the target amount of code is outputted as the error in the amount of code. If the generated amount of code is smaller than the target amount of code, the error is outputted as a positive value. If larger, the error is outputted as a negative value.

[0058]

In this processing to compare amounts of code, the generated amount of code and the target amount of code are compared. In other words, a target is set by deciding on a volume of compressed video data for a transmission path of a predetermined capacity, and

expressed as the target amount of code for each block in the video data. In reality, a certain capacity for data other than the video data is subtracted from the predetermined transmission capacity. The remainder is then divided by the predetermined number of blocks. Naturally, the unit for the data other than video data is the bit.

[0059]

The actual target amount of code inputted to the code amount comparator 6A is set to 78 (integer) bits. By setting an integer number of bits in this way, operating precision is reduced and the scale of the hardware can be suppressed. When 78 bits are used, the target amount of code can be expressed using 7 bit number.

[0060]

As shown in Figure 3A, the accumulator 8A sums (30A, 32A, 34A) the code amount error outputted from the code amount comparator 6A, and outputs data, including an amount of generated code from the past, for producing a scale factor for quantizing the next block. When the amount of generated code for the previous block has not reached the target amount of code, the accumulation value is a large positive value. When the amount of generated code for the previous block has exceeded the target amount of code, the accumulation value is a large negative value. The

accumulation value accumulated in the register 32A is forcibly set in the register 32B on the decoder side every N frames (in the present embodiment, the register 32B is set once every four frames, but N is an integer greater than or equal to 1).

[0061]

As shown in Figure 4, the scale factor calculator 10A is a converting circuit for converting the positive and negative value of the accumulation result to a scale factor which is a positive value. The scale factor is calculated to keep the division data of the divider 2 within an appropriate range. Note that the feedback gain shown in Figure 4 is used to adjust the value of scale factor produced for a predetermined accumulation value, and is input data for adjusting loop gain when controlling the amount of generated code. The offset adjustment is a way of offsetting the positive and negative values of the accumulation value to ensure that all values are positive. The clip processing is used to limit the range taken by the scale factor to a predetermined range while the offsetting continues.

[0062]

The scale factor of the present embodiment is set so that conditions (1) to (3) below are satisfied.

[0063]

(1) The scale factor must be a positive number.

[0064]

(2) Even when quantizing at the finest level, the effective data resulting from the quantization must have no more than a predetermined number of bits.

[0065]

(3) When quantizing at the coarsest level, the effective data must be reduced to zero after the rounding processing.

[0066]

As shown in Figure 9, a data occupancy detector 101 in the occupancy comparator 3 detects write addresses (WA) and read addresses (RA) and the difference between the two numbers (WA - RA), thereby detecting the data occupancy in the buffer memory 14. The adder 102 calculates a difference (positive, negative or zero) between the detected value and a target occupancy (such as half the capacity of the buffer memory 14, or some other value), and outputs the calculated difference.

[0067]

In the accumulator 5, the detected difference values from the occupancy comparator 3 are then accumulated (summed) in a register 104 via an adder 103 and the accumulation value is outputted via a clip processor 105. The register 104 successively adds values from the adder 103 and inputs the summed data (positive or negative) to the clip processor 105 upon

input of a clock signal occurred every 4 frames or the like (4 frame clock). The clip processor 105 clips the inputted data within a range (positive and negative) required by a coefficient multiplier 7 of a next stage, and inputs the clipped data to the coefficient multiplier 7 and feedbacks it to the adder 103.

[0068]

In the coefficient multiplier 7, the accumulation data from the accumulator 5 (positive or negative) which has been limited to the predetermined range is increased in size by a predetermined gain in the multiplier 106, and stored in a register 107. Upon each four frame clock signal, the register 107 inputs the stored data to an adder 25 of the code amount comparator 6A as the correction information amount.

[0069]

The following describes the correction information amount using a specific example. If the actual target amount of code is set to 78 (bits/block) as described above, the gain of the clip processor 105 and the multiplier 106 is adjusted so to give correction information amounts of 0, 1, 2, -1, and -2. This allows output values of 78 (correction information amount = 0), 77 (correction information amount = 1), 76 (correction information amount = 2), 79 (correction information amount = -1), and 80 (correction information amount = -2) to be obtained from the adder

25. If, for instance, the data occupancy of the buffer memory 14 exceeds 1/2, the correction information amount becomes "1" or "2", causing the adder 25 to output "77" or "76". As a result the data occupancy in the buffer memory 14 is gradually reduced. When the data occupancy of the buffer memory 14 falls below 1/2, the reverse of the above-described operation is performed. Thus, the target amount of code is ultimately corrected to within a range of 78 ± 2. The above described operations are also performed in the construction (201 to 207) of on the decoder side, and it is thereby possible for the encoder and the decoder to operate independently. To allow transmission errors and the like to be dealt with, data can be transmitted as initial data from the register 104 on the encoder side to the register 204 on the decoder side every four frames via the transmission path. In this case, it is preferable that the construction (101 to 107) on the encoder side is the same as the construction (201 to 207) on the decoder side.

[0070]

The clock of the registers 104, 107, 204, and 207 is a continuous clock with a period of four frames. Data outputted from the register 104 at a clock timing t_{11} is outputted on the immediately next clock timing t_{12} (= t_{11} + 4 frames) from the register 107. Moreover, data outputted from the register 204 at a clock timing

t_{21} is outputted on the next clock timing t_{22} (= t_{21} + 4 frames) from the register 207. Generally, the timing of t_{11} and t_{21} will be completely unrelated.

[0071]

On the decoder 200 side (see Figure 1), encoded data is inputted from the encoder 100, zeros are inserted based on the zero-run lengths, and data arrays equivalent to the original data arrays are reproduced. Further, processing to calculate amounts of code is performed in the same way as in the encoder (see Figure 7).

[0072]

In the decoder, the constructions used from the processing to calculate amounts of code to generation of the scale factor is exactly the same as in the encoder. This means that it is possible to produce the scale factor from the quantized data without transmitting values for the scale factor from the encoder 100 side to the decoder 200 side. Since the code amount comparators 6A and 6B, the accumulators 8A and 8B, the scale factor calculators 10A and 10B have exactly the same constructions, equivalent inputs result in equivalent outputs. However, if the values in registers 32A and 32B (see Figure 3) of the respective multipliers differ, the respective outputs will also differ. The accumulation value of the encoder 100 is therefore transmitted to the decoder 200

once every four frames to ensure that the values in the registers 32A and 32B match. If a transmission error occurs, it is not subsequently possible to reproduce the scale factor used in the encoder 100 on the decoder side, and decoding will be impossible. The accumulation value used in the decoder is therefore transmitted once every four frames to allow matching of the accumulation results.

[0073]

In the above described embodiment, since a 19 bit accumulation value is transmitted to the decoder once every four frames, the required information capacity for this transmission is 142.5 bit/sec. Since it is necessary to transmit 19 bits of accumulation value data for an image every four frames, the required capacity is:

[0074]

[Equation 1]

$$19 \times (30/4) = 142.5 \text{ (bit/sec)}.$$

On the other hand, when transmitting a scale factor (5 bit) of a high vision signal using the MPEG 2 standard, the required capacity is:

[0075]

[Equation 2]

$$\text{(number of macroblocks)} \times 5 \text{ (bits)} \times 30 \text{ (frame/sec)}$$

$$= ((1920 \times 1036)/(64 \times 4)) \times 5 \times 30 = 116550$$

(bits/sec).

Hence, the present embodiment brings a marked information reduction.

[0076]

Moreover, the present embodiment allows the scale factor (9 bit) of each block to be controlled, effectively enabling realtime control of the blockwise compression ratio. On the other hand, when MPEG 2 is applied to a high vision signal, a scale factor (5 bit) is used for control of each macroblock. From this it is clear that the present embodiment allows a finer gradation in compression control.

[0077]

[Advantages of the Invention]

According to the invention described above, quantization/inverse quantization can be achieved using a simple construction. Moreover, since it is sufficient to transmit accumulation value only once every N frames (where N is an integer greater than or equal to 1), a remarkable improvement in transmission efficiency is possible without sacrificing the data that is originally transmitted.

[0078]

In the case of image data, since it is possible to avoid transmitting a large part of the data for the divisors in the quantization, the extra capacity can be allocated to improve image quality.

[0079]

Furthermore, since it is possible to control the compression ratio for each block, the present invention has a particular advantage with regard to the transmission and recording of highly detailed images.

[0080]

Moreover, when compressing moving picture data for transmission using a high compression ratio, the amount of code taken up by the scale factors of each block in the transmitted data has to be ignored. In this type of case, since transmission of the scale factors can be omitted, the compressed data can be reduced by a corresponding amount to enable high quality data transmission.

[0081]

In addition, if correction of the target amount of code is not performed, the buffer memory occupancy sometimes declines. To prevent such as decline, in some cases null data (dummy data) is inserted. In the present invention, however, correction is performed on the target amount of code, and there is therefore no need to insert null data.

[Brief Description of the Drawings]

[Figure 1]

Figure 1 is a block diagram showing a construction of a first embodiment to which the present

invention is applied.

[Figure 2]

Figure 2 shows a code amount comparator for comparing a target amount of code and an amount of generated code.

[Figure 3]

Figure 3 shows an accumulator.

[Figure 4]

Figure 4 shows a circuit for processing to calculate a scale factor.

[Figure 5]

Figure 5 shows a divider included in an encoder.

[Figure 6]

Figure 6 shows a circuit for performing encoding/code amount calculation processing.

[Figure 7]

Figure 7 shows a circuit for performing decoding/code amount calculation processing.

[Figure 8]

Figure 8 shows a multiplier included in the decoder.

[Figure 9]

Figure 9 is a circuit diagram showing a construction for obtaining a correction information amount.

[Figure 10]

Figure 10 shows an example of a conventional code

amount controlling device.

[Figure 11]

Figure 11 shows a further example of a conventional code input controlling device.

[Description of Symbols]

2 divider

3, 11 occupancy comparator

4 circuit for performing encoding/code amount calculation processing

5, 8A, 8B, 13 accumulator

6A, 6B code amount comparator

7, 15 coefficient multiplier

10A, 10B scale factor calculator

12 Huffman encoding circuit

14 buffer memory

16-1, 16-2, 16-3, 16-4, 16-5 transmission path

18 buffer memory

19 Huffman decoding circuit

20 circuit for performing decoding/code amount calculation processing

22 multiplier

100 encoder

200 decoder

Figure 1

6A	CODE AMOUNT COMPARATOR
6B	CODE AMOUNT COMPARATOR
8A	ACCUMULATOR
8B	ACCUMULATOR
10A	SCALE FACTOR CALCULATOR
10B	SCALE FACTOR CALCULATOR
2	DIVIDER
4	ENCODING/CODE AMOUNT CALCULATION PROCESSING
12	HUFFMAN ENCODING
14	BUFFER MEMORY
18	BUFFER MEMORY
5	ACCUMULATOR
13	ACCUMULATOR
7	COEFFICIENT MULTIPLIER
15	COEFFICIENT MULTIPLIER
3	OCCUPANCY COMPARATOR
11	OCCUPANCY COMPARATOR
100	ENCODER
16-1	TRANSMISSION PATH (VTR)
22	MULTIPLIER
19	HUFFMAN DECODING
20	DECODING/CODE AMOUNT CALCULATION PROCESSING
200	DECODER
#1	ERROR IN AMOUNT OF CODE
#2	ACCUMULATION RESULT
#3	SCALE FACTOR

#4 COMPONENT DATA (INPUT)

#5 DIVIDER DATA

#6 ENCODED DATA (PAIR DATA)

#7 AMOUNT OF CODE

#8 COMPONENT DATA (OUTPUT)

#9 DECODED DATA

Figure 2

6A, 6B CODE AMOUNT COMPARATOR

#1 TARGET AMOUNT OF CODE

#2 ERROR IN AMOUNT OF CODE

#3 CORRECTION INFORMATION AMOUNT

#4 AMOUNT OF GENERATED CODE

Figure 3

32A REGISTER

32B REGISTER

34A CLIP PROCESSING

34B CLIP PROCESSING

#1 ENCODER

#2 ACCUMULATION RESULT

#3 DECODER

Figure 4

10A, 10B SCALE FACTOR CALCULATION PROCESSING

42 CLIP PROCESSING

#1 ACCUMULATION RESULT (ACCUMULATION VALUE OF ERROR IN AMOUNT OF CODE)

#2 SCALE FACTOR

#3 FEEDBACK GAIN

#4 OFFSET ADJUSTMENT

#5 CODE LENGTH ADJUSTMENT GAIN

Figure 5

52Y INVERSE CONVERSION

52U INVERSE CONVERSION

52V INVERSE CONVERSION

56Y ROUNDING PROCESSING

56U ROUNDING PROCESSING

56V ROUNDING PROCESSING

58Y QUANTIZATION TABLE (Y)

58U QUANTIZATION TABLE (U)

58V QUANTIZATION TABLE (V)

#1 DIVISION PROCESSING

#2 SCALE FACTOR

#3 COMPONENT DATA INPUT (Y)

#4 DIVIDER DATA (Y)

#5 COMPONENT DATA INPUT (Cb)

#6 DIVIDER DATA (Cb)

#7 COMPONENT DATA INPUT (Cr)

#8 DIVIDER DATA (Cr)

Figure 6

60Y RUN-LENGTH ENCODING

60U RUN-LENGTH ENCODING

60V RUN-LENGTH ENCODING

62Y AMOUNT OF CODE CALCULATION

62U AMOUNT OF CODE CALCULATION

62V AMOUNT OF CODE CALCULATION

#1 ENCODING/CODE AMOUNT CALCULATION PROCESSING

#2 DIVIDER DATA (Y)

#3 ENCODED DATA (Y)

#4 DIVIDER DATA (Cb)

#5 ENCODED DATA (Cb)

#6 DIVIDER DATA (Cr)

#7 ENCODED DATA (Cr)

#8 AMOUNT OF CODE

Figure 7

70Y RUN-LENGTH DECODING

70U RUN-LENGTH DECODING

70V RUN-LENGTH DECODING

72Y AMOUNT OF CODE CALCULATION

72U AMOUNT OF CODE CALCULATION

72V AMOUNT OF CODE CALCULATION

#1 DECODING/CODE AMOUNT CALCULATION PROCESSING

#2 ENCODED DATA (Y)

#3 DECODED DATA (Y)

#4 ENCODED DATA (Cb)

#5 DECODED DATA (Cb)

#6 ENCODED DATA (Cr)

#7 DECODED DATA (Cr)

#8 AMOUNT OF CODE

Figure 8

82Y QUANTIZATION TABLE (Y)

82U QUANTIZATION TABLE (U)

82V QUANTIZATION TABLE (V)

#1 MULTIPLICATION PROCESSING

#2 SCALE FACTOR

#3 DECODED DATA (Y)

#4 COMPONENT DATA OUTPUT (Y)

#5 DECODED DATA (Cb)

#6 COMPONENT DATA OUTPUT (Cb)

#7 DECODED DATA (Cr)

#8 COMPONENT DATA OUTPUT (Cr)

Figure 9

6A CODE AMOUNT COMPARATOR

6B CODE AMOUNT COMPARATOR

107 REGISTER

207 REGISTER

104 REGISTER

204 REGISTER

105 CLIP PROCESSING

205 CLIP PROCESSING

14 BUFFER MEMORY

18 BUFFER MEMORY

#1 4 FRAME CLOCK

#2 GAIN

#3 TARGET OCCUPANCY (1/2 BUFFER MEMORY)

#4 TO 204

#5 FROM 104

Figure 10

110A QUANTIZER

110B QUANTIZER

110C QUANTIZER

111A VARIABLE LENGTH ENCODER

111B VARIABLE LENGTH ENCODER

111C VARIABLE LENGTH ENCODER

114 OUTPUT SELECTION

112 BUFFER MEMORY

113 JUDGMENT UNIT

#1 IMAGE DATA

#2 INFORMATION ABOUT AMOUNT OF CODE

#3 SELECTION SIGNAL

#4 BUFFER MEMORY OCCUPANCY

Figure 11

120 CODE AMOUNT PREDICTING UNIT

122 QUANTIZER

123 VARIABLE LENGTH ENCODER

121 BUFFER MEMORY

#1 PREDICTION INFORMATION

#2 BUFFER MEMORY OCCUPANCY

#3 IMAGE DATA

(19)日本国特許庁（ＪＰ）　(12) **公開特許公報（A）**　(11)特許出願公開番号

特開平9－98415

(43)公開日　平成9年(1997)4月8日

(51)Int.Cl.[6]		識別記号	庁内整理番号	FI		技術表示箇所
H04N	7/24			H04N	7/13	Z
H03M	7/40		9382－5K	H03M	7/40	
H04N	1/41			H04N	1/41	B

審査請求　未請求　請求項の数13　ＯＬ　（全 15 頁）

(21)出願番号　特願平7－252485

(22)出願日　平成7年(1995)9月29日

(71)出願人　000209751
池上通信機株式会社
東京都大田区池上5丁目6番16号

(72)発明者　今井　英隆
東京都大田区池上5丁目6番16号　池上通信機株式会社内

(72)発明者　水口　穂高
東京都大田区池上5丁目6番16号　池上通信機株式会社内

(72)発明者　原田　裕司
東京都大田区池上5丁目6番16号　池上通信機株式会社内

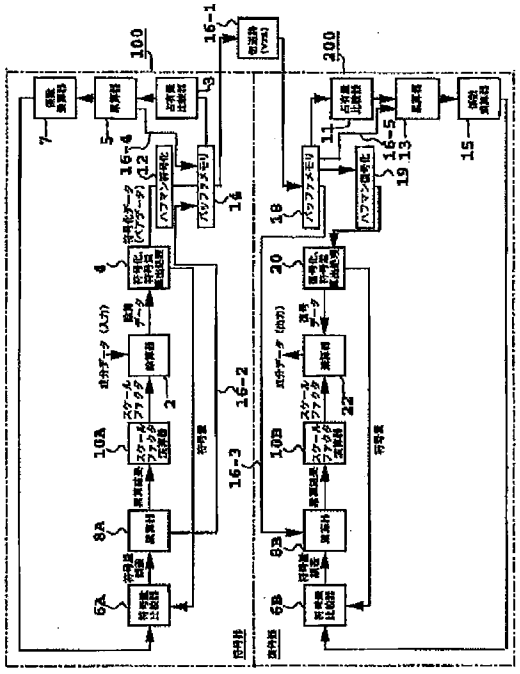
(74)代理人　弁理士　谷　義一　（外1名）

(54)【発明の名称】　量子化／逆量子化を行うための方法および装置

(57)【要約】

【課題】　簡潔な構成で量子化／逆量子化を可能にする。

【解決手段】　占有量比較器3によって、バッファメモリ14の量子化データ占有量を目標占有量と比較し、その差を累算器5で累算し、その累算結果から係数乗算器7で所定範囲内の補正情報を生成し、符号量比較器6Aにフィードバックして、目標符号量を所定範囲内にその変動が収まるように補正する。これによって、目標符号量を整数値とすることができ、ハードウェアの構成が簡潔となる。



1

【特許請求の範囲】

【請求項1】　入力された成分データを量子化して得られた量子化データを伝送する際に、

複数個の前記成分データに対応する1ブロックを一単位として、該1ブロック中に含まれる発生符号量の、予め定められている目標符号量に対する誤差を累算し、

前記目標符号量に対する誤差の累算値を用いて前記発生符号量を制御し、

前記制御後の量子化データを格納するバッファメモリにおける前記量子化データの占有量の、予め定められている目標占有量に対する誤差を累算し、

前記目標占有量に対する誤差の累算値を用いて前記目標符号量を補正することを特徴とする量子化を行うための方法。

【請求項2】　請求項1において、前記制御に際しては、

前記目標符号量に対する誤差の累算値に所定の演算処理を施すことによりスケールファクタを算出し、各ブロック毎に予め設定されている量子化テーブル値に前記スケールファクタを乗算することにより、量子化のための前記除数とし、

Nフレーム（Nは、1以上の整数）毎にのみ、前記目標符号量に対する誤差の累算値を逆量子化装置側へ送出することを特徴とする量子化を行うための方法。

【請求項3】　請求項2において、前記目標占有量に対する誤差の累算値をNフレームごとにのみ逆量子化装置側へ送出することを特徴とする量子化を行うための方法。

【請求項4】　請求項2において、前記発生符号量を算出する際には、1ブロックM個（Mは1以上の整数）のデータを代表するゼロラン長と非ゼロデータとから成るペアデータに基づいて、符号化されたビット数を算出することを特徴とする量子化を行うための方法。

【請求項5】　量子化装置側から伝送されてきた量子化符号データに所定の乗数を乗算することにより、逆量子化を行うにあたり、

前記量子化装置側から伝送されてきた量子化符号データを格納するバッファメモリからの前記量子化符号データに基づく複数個の成分データに対応する1ブロック毎の発生符号量の、前記量子化装置側で使用されている目標符号量に対する誤差を累算し、

前記目標符号量に対する誤差の累算値を用いて乗数を生成し、

前記バッファメモリにおける前記量子化データの占有量の、予め定められている目標占有量に対する誤差を累算し、

前記目標占有量に対する誤差の累算値を用いて前記目標符号量を補正することを特徴とする逆量子化を行うための方法。

【請求項6】　請求項5において、さらに、

2

前記目標符号量に対する誤差を累算するに際して、Nフレーム（Nは1以上の整数）毎に前記量子化装置側から伝送されて来る累算値と等しくなるよう当該目標符号量に対する誤差の累算値を整定し、

前記累算値に対して、前記量子化装置側と同様の演算処理を施すことによりスケールファクタを算出し、

前記量子化装置側で用いられている量子化テーブルと同一の値を有する逆量子化テーブル値に、前記スケールファクタを乗算することにより、逆量子化のための前記乗数とすることを特徴とする逆量子化を行うための方法。

【請求項7】　請求項6において、前記発生符号量を算出する際には、ゼロランレングス復号化により得られるゼロラン長と非ゼロデータとから成るペアデータに基づいて、符号化されたビット数を算出することを特徴とする逆量子化を行うための方法。

【請求項8】　入力された成分データを量子化して得られた量子化データを伝送する量子化装置であって、

複数個の前記成分データに対応する1ブロックを一単位として、該1ブロック中に含まれる発生符号量の、予め定められている目標符号量に対する誤差を累算して累算値を得る第1累算手段と、

前記目標符号量に対する誤差の累算値を用いて前記発生符号量を制御する制御手段と、

前記制御後の量子化データを格納するバッファメモリと、

前記バッファメモリにおける前記量子化データの占有量の、予め定められている目標占有量に対する誤差を累算して累算値を得る第2累算手段と、

前記目標占有量に対する誤差の累算値を用いて前記目標符号量を補正する補正手段とを具えたことを特徴とする量子化装置。

【請求項9】　請求項8において、前記制御手段は、

前記目標符号量に対する誤差の累算値に所定の演算処理を施すことによりスケールファクタを算出するスケールファクタ演算手段と、

各ブロック毎に予め設定されている量子化テーブル値に前記スケールファクタを乗算することにより、量子化のための前記除数を算出する除数演算手段と、

Nフレーム（Nは、1以上の整数）毎にのみ、前記累算値を逆量子化装置側へ送出する累算値伝送手段とを有することを特徴とする量子化装置。

【請求項10】　請求項9において、前記発生符号量は、1ブロックM個（Mは1以上の整数）のデータを代表するゼロラン長と非ゼロデータとから成るペアデータに基づいて、符号化されたビット数が算出されていることを特徴とする量子化装置。

【請求項11】　量子化装置側から伝送されてきた量子化符号データに所定の乗数を乗算することにより、逆量子化を行う逆量子化装置であって、

前記量子化装置側から伝送されてきた量子化符号データ

を格納するバッファメモリと、
前記バッファメモリからの前記量子化符号データに基づく複数個の成分データに対応する1ブロック毎の発生符号量の、前記量子化装置側で使用されている目標符号量に対する誤差を累算して累算値を得る第1累算手段と、
前記目標符号量に対する誤差の累算値を用いて乗数を生成する生成手段と、
前記バッファメモリにおける前記量子化データの占有量の、予め定められている目標占有量に対する誤差を累算して累算値を得る第2累算手段と、
前記目標占有量に対する誤差の累算値を用いて前記目標符号量を補正する補正手段とを具えたことを特徴とする逆量子化装置。

【請求項12】　請求項11において、前記乗数生成手段は、
前記目標符号量に対する誤差を累算する累算回路と、Nフレーム（Nは1以上の整数）毎に前記量子化装置側から伝送されて来る累算値を導入することにより、該累算回路の累算値を整定する累算値整定回路とを有する累算手段と、
前記累算手段から得られた累算値に対して、前記量子化装置側と同様の演算処理を施すことによりスケールファクタを算出するスケールファクタ演算手段と、
前記量子化装置側で用いられている量子化テーブルと同一の値を有する逆量子化テーブル値に、前記スケールファクタを乗算することにより、逆量子化のための前記乗数を算出する乗数演算手段とを有することを特徴とする逆量子化装置。

【請求項13】　請求項12において、前記発生符号量は、ゼロランレングス復号化により得られるゼロラン長と非ゼロデータとから成るペアデータに基づいて、符号化されたビット数が算出されることを特徴とする逆量子化装置。

【発明の詳細な説明】

【0001】

【発明の属する技術分野】本発明は、画像データなどの圧縮伝送を行うのに好適な、量子化／逆量子化を行うための方法および装置に関するものである。

【0002】

【従来の技術】まず、量子化および逆量子化について、一般的に知られている技術を述べる。

【0003】量子化とは、成分出力値をその成分に固有の数値で除算し、小数点を四捨五入することである。四捨五入されるため逆量子化時にその数値を乗算しても、完全には復元されない。圧縮された画像の画質と圧縮率は、量子化の処理で決定される。画質を良くするためには、成分出力値を小さな値で割れば良く、この場合圧縮率は低くなる。逆に大きな値で割れば、有効ビット数が減少するため、圧縮率は大きくなるものの画質は低下する。また量子化において、一定の数値で割れば、入力画

像によって符号量は変化する。

【0004】そこで、入力画像によって除算する数値を変え、出力符号量を一定に保つように制御することが考えられる。また、量子化と逆量子化では、除数と乗数は同じ数値でなければならない。そのために、量子化器で使用した数値（スケールファクタ）を逆量子化器側に伝送することが行われている。

【0005】ここで、画像符号化における符号量制御を例にとり従来技術を説明する。

【0006】最も単純な例として、たとえば図10に示すように複数の量子化器110A～110Cおよび可変長符号化器111A～111Cとバッファメモリ112を用意する。そして、バッファメモリ112の占有量の目標値に対する誤差の累算値に応じて、判定部113によって、出力選択器114を制御して適切な符号量をもつ量子化器の出力データを選択する。

【0007】あるいは、たとえば図11のように、符号量予測部120において、画像データから何らかの方法を用いて（たとえば空間的情報や周波数的情報など）符号量を予測し、その結果とバッファメモリ121の占有量とから量子化器122を制御し、可変長符号化器123に制御後の量子化データを送出する方法がある。

【0008】また、上記のスケールファクタは、量子化テーブルの全体にかかる乗率であって、スケールファクタの値を適宜変更していくことにより、伝送線路での情報量が一定になるように制御される。

【0009】すなわち、スケールファクタを伝送路上に送出する理由は、符号器で使用したスケールファクタの値が判らないと、復号器では、どの値で割ったのか判らないため、復号することができないからである。なお、符号器でスケールファクタを使用しない場合には（一定のスケールファクタ値を使う場合には）、伝送線路の情報量は入力データに依存することになるので、伝送線路の情報量を一定にすることができない。

【0010】一方、動画像の圧縮方式として、MPEG2の国際標準方式がある。この規格（ISO／IEC 13818－2 “Information Technology－Generic coding of moving pictures and associated audio information－Draft Internaional Standard 1994”）によれば、マクロブロック単位でスケールファクタを変えることができ、Y：Cb：Cr＝4：2：2の符号化では、Yで4ブロック、Cbで2ブロック、Crで2ブロックの合計8ブロックからなるマクロブロックに対して、5ビットを割り当てている。

【0011】MPEG2においてスケールファクタ以外の情報を全て無視したとすれば、ハイビジョン信号においては、1920（水平画素数）×1036（垂直画素数）×5（ビット）×30（フレーム／秒）×2（4：

2：2方式）／64（1ブロックの画素数）／8（ブロック）＝1165500（bit／sec）という膨大な情報を送る必要がある。

【0012】

【発明が解決しようとする課題】上述したように、従来の技術における問題点として、図10の構成をとる場合、制御の精度を向上させるためには量子化器を多数用意しなければならず、ハードウェアの規模が多くなる。これを解決する方法として図11の構成をとると、符号量を予測する複雑な機構が必要となり、やはりハードウェアの規模は小さくならない。

【0013】また、従来のスケールファクタを伝送する方式では、限られた容量の伝送線路上に多量のスケールファクタ情報を送出しなければならなかった。

【0014】しかも、伝送すべき画像が高精細になるほど、スケールファクタの伝送に要する情報量は極めて大きな値になるという問題があった。

【0015】よって本発明の目的は上述の点に鑑み、複数の量子化器あるいは符号量予測のような特別な機構を必要とせずに、簡潔な構成を用いて量子化／逆量子化を可能とし、さらに、スケールファクタの伝送に要する情報量を省略し、あるいは従来に比して激減させることにより、所要データの伝送効率の向上を図った、量子化／逆量子化を行うための方法および装置を提供することにある。

【0016】

【課題を解決するための手段】かかる目的を達成するために、請求項1にかかる発明は、入力された成分データを量子化して得られた量子化データを伝送する際に、複数個の前記成分データに対応する1ブロックを一単位として、該1ブロック中に含まれる発生符号量の、予め定められている目標符号量に対する誤差を累算し、前記目標符号量に対する誤差の累算値を用いて前記発生符号量を制御し、前記制御後の量子化データを格納するバッファメモリにおける前記量子化データの占有量の、予め定められている目標占有量に対する誤差を累算し、前記目標占有量に対する誤差の累算値を用いて前記目標符号量を補正することを特徴とする。

【0017】また、請求項2にかかる発明は、請求項1において、前記制御に際しては、前記目標符号量に対する誤差の累算値に所定の演算処理を施すことによりスケールファクタを算出し、各ブロック毎に予め設定されている量子化テーブル値に前記スケールファクタを乗算することにより、量子化のための前記除数とし、Nフレーム（Nは、1以上の整数）毎にのみ、前記目標符号量に対する誤差の累算値を逆量子化装置側へ送出することを特徴とする。

【0018】さらに、請求項3にかかる発明は、請求項2において、前記目標占有量に対する誤差の累算値をNフレームごとにのみ逆量子化装置側へ送出することを特徴とする。

【0019】さらに、請求項4にかかる発明は、請求項2において、前記発生符号量を算出する際には、1ブロックM個（Mは1以上の整数）のデータを代表するゼロラン長と非ゼロデータとから成るペアデータに基づいて、符号化されたビット数を算出することを特徴とする。

【0020】さらに、請求項5にかかる発明は、量子化装置側から伝送されてきた量子化符号データに所定の乗数を乗算することにより、逆量子化を行うにあたり、前記量子化装置側から伝送されてきた量子化符号データを格納するバッファメモリからの前記量子化符号データに基づく複数個の成分データに対応する1ブロック毎の発生符号量の、前記量子化装置側で使用されている目標符号量に対する誤差を累算し、前記目標符号量に対する誤差の累算値を用いて乗数を生成し、前記バッファメモリにおける前記量子化データの占有量の、予め定められている目標占有量に対する誤差を累算し、前記目標占有量に対する誤差の累算値を用いて前記目標符号量を補正することを特徴とする。

【0021】さらに、請求項6にかかる発明は、請求項5において、さらに、前記目標符号量に対する誤差を累算するに際して、Nフレーム（Nは1以上の整数）毎に前記量子化装置側から伝送されて来る累算値と等しくなるよう当該目標符号量に対する誤差の累算値を整定し、前記累算値に対して、前記量子化装置側と同様の演算処理を施すことによりスケールファクタを算出し、前記量子化装置側で用いられている量子化テーブルと同一の値を有する逆量子化テーブル値に、前記スケールファクタを乗算することにより、逆量子化のための前記乗数とすることを特徴とする。

【0022】さらに、請求項7にかかる発明は、請求項6において、前記発生符号量を算出する際には、ゼロランレングス復号化により得られるゼロラン長と非ゼロデータとから成るペアデータに基づいて、符号化されたビット数を算出することを特徴とする。

【0023】さらに、請求項8にかかる発明は、入力された成分データを量子化して得られた量子化データを伝送する量子化装置であって、複数個の前記成分データに対応する1ブロックを一単位として、該1ブロック中に含まれる発生符号量の、予め定められている目標符号量に対する誤差を累算して累算値を得る第1累算手段と、前記目標符号量に対する誤差の累算値を用いて前記発生符号量を制御する制御手段と、前記制御後の量子化データを格納するバッファメモリと、前記バッファメモリにおける前記量子化データの占有量の、予め定められている目標占有量に対する誤差を累算して累算値を得る第2累算手段と、前記目標占有量に対する誤差の累算値を用いて前記目標符号量を補正する補正手段とを具えたことを特徴とする。

【００２４】さらに、請求項９にかかる発明は、請求項８において、前記制御手段は、前記目標符号量に対する誤差の累算値に所定の演算処理を施すことによりスケールファクタを算出するスケールファクタ演算手段と、各ブロック毎に予め設定されている量子化テーブル値に前記スケールファクタを乗算することにより、量子化のための前記除数を算出する除数演算手段と、Ｎフレーム（Ｎは、１以上の整数）毎にのみ、前記累算値を逆量子化装置側へ送出する累算値伝送手段とを有することを特徴とする。

【００２５】さらに、請求項１０にかかる発明は、請求項９において、前記発生符号量は、１ブロックＭ個（Ｍは１以上の整数）のデータを代表するゼロラン長と非ゼロデータとから成るペアデータに基づいて、符号化されたビット数が算出されていることを特徴とする。

【００２６】さらに、請求項１１にかかる発明は、量子化装置側から伝送されてきた量子化符号データに所定の乗数を乗算することにより、逆量子化を行う逆量子化装置であって、前記量子化装置側から伝送されてきた量子化符号データを格納するバッファメモリと、前記バッファメモリからの前記量子化符号データに基づく複数個の成分データに対応する１ブロック毎の発生符号量の、前記量子化装置側で使用されている目標符号量に対する誤差を累算して累算値を得る第１累算手段と、前記目標符号量に対する誤差の累算値を用いて乗数を生成する生成手段と、前記バッファメモリにおける前記量子化データの占有量の、予め定められている目標占有量に対する誤差を累算して累算値を得る第２累算手段と、前記目標占有量に対する誤差の累算値を用いて前記目標符号量を補正する補正手段とを具えたことを特徴とする。

【００２７】さらに、請求項１２にかかる発明は、請求項１１において、前記乗数生成手段は、前記目標符号量に対する誤差を累算する累算回路と、Ｎフレーム（Ｎは１以上の整数）毎に前記量子化装置側から伝送されて来る累算値を導入することにより、該累算回路の累算値を整定する累算値整定回路とを有する累算手段と、前記累算手段から得られた累算値に対して、前記量子化装置側と同様の演算処理を施すことによりスケールファクタを算出するスケールファクタ演算手段と、前記量子化装置側で用いられている量子化テーブルと同一の値を有する逆量子化テーブル値に、前記スケールファクタを乗算することにより、逆量子化のための前記乗数を算出する乗数演算手段とを有することを特徴とする。

【００２８】さらに、請求項１３にかかる発明は、請求項１２において、前記発生符号量は、ゼロランレングス復号化により得られるゼロラン長と非ゼロデータとから成るペアデータに基づいて、符号化されたビット数が算出されることを特徴とする。

【００２９】

【発明の実施の形態】符号器（図１：１００）側では、一つのブロックの伝送符号量を計数し、その計数結果とブロックの目標符号量との差を連続累算し、その累算値の値に所定の係数を乗じ、所定の係数を加算し、次のブロック量子化にあたってのスケールファクタを決定する方法をとる。さらに、伝送すべきデータを格納するバッファメモリにおけるデータの占有量が目標占有量になるように目標符号量を補正する。

【００３０】復号器（図１：２００）側では、符号器側におけるスケールファクタの決定動作における所定係数を持ち、符号器と同じ動作をする回路を設け、伝送されてきたブロックの符号量を計測し、その量を符号器側と同様の回路に供給して、符号器側の同一のブロックの目標符号量との差を連続累算し、その累算値の値に応じて、次のブロックの逆量子化にあたってのスケールファクタを決定する方法をとる。さらに、伝送されてきたデータを格納するバッファメモリにおけるデータの占有量が所定の量になるように目標符号量を補正する。

【００３１】以上の構成によれば、簡潔な構成で量子化／逆量子化が可能となり、さらに、以上の構成を採り、双方の初期累算値を同一にすれば、以後の動作は全く同一に成り、同じブロックに対して同じスケールファクタ値が得られる。言い換えれば、累算値が同じであれば、同じスケールファクタ値が得られる。その後のブロックにおいても、そのブロック符号量が同じあれば累算値も同じに成る。

【００３２】従って、基本的には符号器と復号器との累算値の初期化は最初の時点で行うだけで、以後のスケールファクタの一致が行える。しかしながら、符号器と復号器とは、通常は別の場所にあり、また同時に電源ＯＮが行われることも考えられない。互いに無関係に運用されても双方の累算値の一致化が瞬時に行われるためには、常時その一致化が行われることが必要になる。実施例では、処理単位として４フレーム毎に累算値を伝送するようにしている。当然、その間における個々のフレーム内のブロック毎のスケールファクタの一致化は、ブロックの符号量を通じて行われる。このことは、ブロック単位にスケールファクタを任意に変化させることができるので、画像に応じたきめ細かい符号量制御が可能に成る。また、ブロック単位に行うので、多数のブロックで構成される１フレームにおいては、事前にそのフレーム全体の画像の調査をしなくとも、比較的適切な符号量に設定することが可能になる。

【００３３】上記の実施の態様を、より具体的に述べると、以下の通りである。

【００３４】１．スケールファクタの決定を行うためには、各ブロックの目標符号量と、計数されたブロック符号量の差をとりブロック毎に累算する。この累算値に所定係数を乗算し、所定係数を加算して、次のブロックの量子化をする際のスケールファクタを算出する。このための構成は、符号器と復号器で同じであり、目標符号量

と所定係数も符号器と復号器で同じに設定する。ただし、符号器では該累算器の値を出力する機能を持ち、復号器では、符号器からの累算値を入力し累算器の値をセットする機能を持っている。

【００３５】２．復号器側では、伝送されてくる連続データから、ブロック毎の符号量を計数する。計数された符号量は、上述のスケールファクタ決定回路に供給される。

【００３６】３．復号器側では、符号器から送られる累算値を受けてスケールファクタ決定回路に供給し、スケールファクタ決定回路の累算器の状態をその値に設定する。このことで、復号器におけるその符号量が計数された次のブロックのスケールファクタは、符号器と同じ値になる。

【００３７】４．１ブロックの情報量とそのブロックの目標符号量との差をブロック毎に累算し、そのブロック毎の累算結果から所定係数を乗算し、所定係数を加算し、スケールファクタを決定している。累算値が同一で、乗算する係数、加算する係数が同一であれば、得られるスケールファクタは同一になる。

【００３８】１フレーム分データの最初に、前のフレームの最後のブロック符号量を累算した結果としての累算値データが復号器側に送られてくる。符号器側は、この累算値データを初期値として次のフレームの処理が行われる。復号器側も送られてきた同じデータを累算値の初期値データとして以後に続く次のフレームの処理を行う。後に続くフレームの処理では、１ブロックの処理が完了する毎に、そのブロックの符号量算出結果で累算値の更新が行われ、新しい累算値は次のブロックの量子化のスケールファクタを導き出す。

【００３９】復号器において、あるブロックが伝送され、ハフマン復号されたランレングス符号のペアデータから符号器と同じようにそのペアデータの符号長を得て、それをブロック単位で累算して符号長を算出し、その算出結果がスケールファクタ決定回路に送られると、その時点での累算値が決定される。言い換えれば、次のブロックの逆量子化にあたってのスケールファクタが決定される。

【００４０】複数ブロックが伝送されてフレームとしての伝送が完了した段階で、次のフレーム処理に移る。４フレーム毎に、符号器側の累算値を伝送路を介して復号器側に送り、復号器側では、送られた累算値をセットすることにより、その累算値に基づいたスケールファクタで次の最初のブロックの量子化が行われる。

【００４１】５．符号器側では伝送すべき量子化データを格納するためのバッファメモリのデータ占有量の、目標占有量に対する誤差を累算し、この累算値を用いて例えば４フレーム毎に目標符号量を補正することによって、バッファメモリにおけるデータ占有量が目標量になるようにする。復号器側においても同様である。

【００４２】以上述べた方法で符号器と復号器が動作した場合、フレームの最初の時点で、累算値が一致していれば、以後の続くブロックの量子化と逆量子化は、同じスケールファクタで行われることになる。基本的には、最初の時点で累算値の一致化が行われていれば、以後は伝送路に障害がない限り、伝送したデータが正確に復号器に受け取られ、復号器では、符号器で使用したスケールファクタで逆量子化が行われることになる。

【００４３】伝送線路の状態が悪く、伝送エラー等がある場合、符号量の計測において、符号器と復号器との間で違いが生じる。このような状態になった場合にも、本発明では、間欠的に累算値を送ることにより、定期的に整定を行っているので、異常状態は短時間にとどまり、符号器と復号器との間でスケールファクタの一致化が成されることになる。

【００４４】

【実施例】以下、図面を参照して、本発明の一実施例を詳細に説明する。

【００４５】図１は、符号器と復号器における本発明を適用した量子化器の部分のブロックダイヤグラムを示した図である。この図では、本発明に関係しない部分、特に量子化前の映像処理部分等は省いている。量子化そのものは除算器２に相当するブロックで行われる。本図において、一点鎖線で囲んだ枠１００は符号器である。この符号器１００に含まれる各ブロックのうち、２は画像データである成分データを入力する除算器であって、その詳細な回路構成は図５に示す。４は符号化・符号量算出処理を行う回路であって、その詳細な回路構成は図６に示す。６Ａは符号量比較器であって、その詳細な回路構成は図２に示す。８Ａは累算器であって、その詳細な回路構成は図３の（Ａ）に示す。１０Ａはスケールファクタ演算器であって、その詳細な回路構成は図４に示す。

【００４６】また、１２は符号化データ（ペアデータ）を入力するハフマン符号化回路、１４は伝送速度を一定に保つためのバッファメモリである。

【００４７】３は占有量比較器、５は累算器、７は係数乗算器であって、これらの詳細な回路構成は図９に示す。

【００４８】この符号器１００に接続されている１６－１は、デジタル形式の伝送路である。この伝送路１６－１は、デジタルデータの記録再生器、デジタルＶＴＲなどとすることも可能である。１６－２（および１６－３）は、本発明における目標符号量に対する誤差の累算値を符号器と復号器とで一致化するためのデータラインを示す。また、１６－４（および１６－５）は本発明における目標符号量に対する誤差の累算値を符号器と復号器とで一致化するためのデータラインを示す。これらのデータは、符号化された画像データと共に多重化されて伝送路１６－１に送り出される。

11

【００４９】符号器１００の下方に示した一点鎖線の枠２００は、復号器を示す。この復号器２００に含まれている６Ｂは符号量比較器であって、符号器側の符号量比較器６Ａ（図２参照）と同一の構成を有する。８Ｂは累算器であって、その詳細な回路構成は図３の（Ｂ）に示す。１０Ｂはスケールファクタ演算器であって、符号器側のスケールファクタ演算器１０Ａ（図４参照）と同一の構成を有する。１８はバッファメモリ、１９はハフマン復号化回路である。２０は復号化・符号量算出処理を行う回路であって、その詳細な回路構成は図７に示す。２２は復号された成分データを出力する乗算器であって、その詳細な回路構成は図８に示す。

【００５０】１１は占有量比較器、１３は累算器、１５は係数乗算器であって、これらの詳細構成は図９に示す。

【００５１】次に、図１ならびに図２～図９を参照して、本実施例の動作を詳細に説明していく。

【００５２】まず、除算器２に入力される成分データは、画像データを直交変換し、周波数成分に分解されたデータ（１２ビット）である。データフォーマットは、２の補数で、１２ビット精度であり、６４個のデータで１つのブロックを構成している。例えば、ハイビジョン信号では１フレーム当たり１９２０×１０３６／６４＝３１０８０個のブロックからなる。

【００５３】除算器２では、入力された成分データを、（スケールファクタ×量子化テーブルの各要素値）で除算し、その後、丸め処理が行われる（図５参照）。丸め＊

12

＊て四捨五入されると、復号器では成分データが完全には元に戻らないことになるため、画質に影響する。すなわち、成分データを大きな数で割れば、０の数が増えるため、圧縮率は大きくなるが画質が低下する。

【００５４】次段の符号化・符号量算出処理回路４では、図６に示すように、ランレングス符号化（６０Ｙ，６０Ｕ，６０Ｖ）を行い、かつ１ブロックの符号量を算出する（６２Ｙ，６２Ｕ，６２Ｖ，６４，６６）。このランレングス符号化では、除算データからゼロの長さ（ゼロランレングス）と非ゼロデータのペアデータに分離する。１ブロックの符号量の算出は、ハフマン符号化された後の１ブロック毎のビット長の算出を、ペアデータの段階でそれぞれのペアデータに相当するビット長をブロック毎に積算することで行っている。この符号量算出は、ハフマン符号化と同時処理で行っている。

【００５５】すなわち、符号量算出処理では、次の表１に示すハフマンテーブルを参照して、１ブロックの合計のビット長を算出する。表１において、例えば、１／２のペアデータの場合は、ゼロラン長が１で、非ゼロデータが２であり、生成されるビット長は６ビットであることを示している。このほかに、非デロデータの絶対値を示す付加ビットを２ビット伝送する。成分データのデータ長は、１２ビットなので１／２というペアデータが生成された場合、（６＋２）／（１２＋１２）＝１／３にデータ圧縮されたことになる。

【００５６】

【表１】

[Table 1]

① ハフマンテーブルの例

②ペアデータ	③ビット長	④生成コード	ペアデータ	ビット長	生成コード
0/0(EOB)	2	00			
0/1	2	01	1/1	4	1011
0/2	3	100	1/2	6	111001
0/3	4	1010	1/3	8	11110110
0/4	5	11000	1/4	9	111110101
0/5	5	11001	1/5	11	11111101100
0/6	6	111000	1/6	12	111111000000
0/7	7	1111000	1/7	12	111111000110
0/8	9	111110100	1/8	12	111111001000

【００５７】符号量比較器６Ａでは、図２に示すように、加算器２５からの補正情報量との差分（詳細後述）を算出した目標符号量と、発生情報量とを加算器２６で比較する。ここで発生情報量とは、その前のブロック処理後に更新された累算器８Ａのデータによって決定されたスケールファクタを使用して量子化され、ランレングス符号化され、そしてハフマン符号化された１ブロックのデータ量であり、ビット単位で表現されている。また、目標符号量とは、１ブロック単位当たりの目標符号量であり、ビット単位で表現されている。目標符号量から発生した符号量を減算した結果が符号量誤差として出力される。ここで、発生符号量が目標符号量よりも少なければ、その誤差は正の値として出力され、逆であれば、負の値として出力される。

【００５８】このように符号量比較処理では、発生した符号量と目標符号量を比較する。換言すれば、所定の伝送路の容量に対して、圧縮映像データとしてどの程度の容量とすべきかの目標を与えるものであり、それを映像のブロック単位で表現している。実際には、所定の伝送容量から、映像データ以外のデータ量を減算し、その結果を所定のブロック数で除算した数値である。当然、このデータの単位はビット単位である。

【００５９】さらに、符号量比較器６Ａに入力する実際の目標符号量は７８（整数）ビットとする。このように、整数ビットにすることによって、演算精度を下げて、ハードウェアの規模を抑えることができる。７８ビットであれば、７ビットで表現できる。

【００６０】累算器８Ａは、図３の（Ａ）に示すように、符号量比較器６Ａから出力される符号量誤差を累積加算（３０Ａ，３２Ａ，３４Ａ）し、過去からの発生情報量を含めて、次のブロックを量子化する際のスケールファクタを作成するデータを作り出す。過去から引き続いてブロックの発生符号量が目標符号量に達していないとき、累算値は大きな正の値になっている。また、過去から引き続いて発生符号量が目標符号量を超えているとき、累算値は負の大きな値になっている。また、レジスタ３２Ａに蓄積されている累算値はＮフレーム毎に（本実施例では４フレーム毎に１回：Ｎは１以上の整数）、復号器側のレジスタ３２Ｂへ強制的にセットされる。

【００６１】スケールファクタ演算器１０Ａは、図４に示すように、正負の値をもつ累算結果から、正の数値であるスケールファクタの値に変換するための変換回路であり、除算器２の除算データが適切な範囲に収まるように演算する。なお、図４に示したフィードバックゲインとは、所定の累算値に対してどのスケールファクタ値を与えるかの調整であり、発生符号量を帰還制御する際のループゲイン調整のための入力データである。オフセット調整とは正負の値を持つ累算値をオフセットして全て正の値にするための措置であり、クリップ処理は、この措置を行いつつ、スケールファクタの採り得る範囲を所定の範囲内に制限する目的を持っている。

【００６２】また、本実施例においては、スケールファクタとして以下の条件①～③を満足するものとする。

【００６３】①スケールファクタの値は、正の数であること。

【００６４】②最も細分化した量子化を行った場合でも、量子化された有効データが所定のビット数以下であること。

【００６５】③最も粗く量子化を行った場合、丸め処理を行った後では、有効データは無くゼロになること。

【００６６】図９に示すように、占有量比較器３においては、データ占有量検出器１０１が、バッファメモリ１４のライトアドレス（ＷＡ）とリードアドレス（ＲＡ）を検出し、両者の差（ＷＡ－ＲＡ）、すなわち、バッファメモリ１４内のデータ占有量を検出し、この検出値と、目標占有量（例えばバッファメモリ１４の容量の１／２。他の値でもよい）との差（正または負または０）を加算器１０２が算出し、出力する。

【００６７】ついで、累算器５においては、占有量比較器３からの差検出値を加算器１０３を介してレジスタ１０４に累算（蓄積）し、クリップ処理器１０５を介して出力する。レジスタ１０４は、加算器１０３からの値を逐次蓄積し、例えば４フレーム毎に発生するクロック（４フレームクロック）の入力時に蓄積されているデータ（正，負）をクリップ処理器１０５に入力する。クリップ処理器１０５は、この入力データを後段の係数乗算器７が要求する範囲内（正，負）にクリップし、係数乗算器７に入力すると共に加算器１０３にフィードバックする。

【００６８】係数乗算器７においては、累算器５からの所定範囲内に制限された累算データ（正，負）を乗算器１０６で所定ゲインにレベルアップし、レジスタ１０７に格納する。レジスタ１０７は４フレームクロック毎に格納データを補正情報量として符号量比較器６Ａの加算器２５に入力する。

【００６９】ここで、補正情報量について具体例を挙げて説明する。前述したように実際の目標符号量を７８（ビット／ブロック）とすると、補正情報量としては、０，１，２，－１，－２が形成されるように、クリップ処理器１０５および乗算器１０６のゲインを調整する。これによって、加算器２５の出力値としては、７８（補正情報量０），７７（同１），７６（同２），７９（同－１），８０（同－２）が得られることになる。例えば、バッファメモリ１４のデータ占有量が１／２を越えてくると、補正情報量が１または２となり、その結果、加算器２５の出力は７７または７６となるので、バッファメモリ１４内のデータ占有量が徐々に減っていくことになる。バッファメモリ１４のデータ占有量が１／２より減ってくると上述と逆の動作となり、結局、目標符号量は７８±２の範囲内に補正制御される。以上の動作は、復号器側の構成（２０１～２０７）においても同様であり、符号器側と復号器側とで独立して動作することができる。なお、伝送エラーが生じた時等に対応するため、符号器側のレジスタ１０４から復号器側のレジスタ２０４に伝送路を介して４フレーム毎にデータをイニシャルデータとして与えることもできる。ただし、この場合、符号器側の構成（１０１～１０７）と復号器側の構成（２０１～２０７）とは同一であることが望ましい。

【００７０】なお、レジスタ１０４，１０７，２０４，２０７のクロックは、いずれのクロックも４フレーム周期で連続したクロックである。レジスタ１０４において、クロックタイミング t_{11} において出力されたデータは、その直後のクロックタイミング t_{12}（＝t_{11}＋４フレーム）においてレジスタ１０７から出力される。また

レジスタ２０４において、t_{21} のタイミングで出力されたデータは、t_{22} （＝t_{21} ＋４フレーム）においてレジスタ２０７から出力される。一般に t_{11} と t_{21} のタイミング関係は全く無関係である。

【００７１】一方、復号器２００（図１参照）側では、符号器１００から送られてきた符号化データを入力し、ゼロラン長に基づき「０」のデータを挿入して、元のデータ列と同じデータ列を再生する。また、符号量算出処理は、符号器と同じ方法で算出する（図７参照）。

【００７２】復号器では、符号量算出処理からスケールファクタの生成まで符号器と全く同じ構成を採っている。このことは、スケールファクタの値を符号器１００側から復号器２００側へ送らなくても、量子化データからスケールファクタを作成できることを意味している。すなわち符号量比較器６Ａと６Ｂ、累算器８Ａと８Ｂの累算回路、スケールファクタ演算器１０Ａと１０Ｂは全く同じ構成であるため、入力が等しければ、出力も等しくなる。ただし、乗算器のレジスタ３２Ａ，３２Ｂ（図３参照）の値が違うと出力も違ってくるので、４フレーム＊に１回だけ符号器１００の累算値を復号器２００に送り、両レジスタ３２Ａ，３２Ｂの値を一致させている。すなわち、伝送エラーが生じると、それ以降は符号器１００で使用したスケールファクタを復号器側で再現できず、復号できなくなってしまうので、４フレームに１回だけ符号器で使用した累算値を送出し、累算結果を一致させている。

【００７３】上述した実施例においては、４フレーム毎に１回だけ１９ビットの累算値を復号器側へ伝送するので、これに要する情報量は１秒当たり１４２．５ビットとなる。すなわち、４フレームの画像に対して１９ビットの累算値データを送る必要から、

【００７４】

【数１】19×（30/4）＝142.5（ビット／秒）

となる。他方、ハイビジョン信号のスケールファクタ（５ビット）をＭＰＥＧ２規格で伝送しようとすると、

【００７５】

【数２】

（マクロブロックの数）×５（ビット）×30（フレーム／秒）
＝（（1920×1036）／（64×４））×５×30
＝116550（ビット／秒）

が必要となるので、本実施例による情報削減の効果は顕著なものとなる。

【００７６】しかも本実施例では、１ブロック毎にスケールファクタ（９ビット）を制御しているので、これは、１ブロック単位で圧縮率をリアルタイム制御していることにほかならない。他方、ハイビジョン信号についてＭＰＥＧ２を適用した場合には、１マクロブロック毎にスケールファクタ（５ビット）を制御することになる。このことから明らかなように、本実施例によれば、より細かな圧縮制御が可能となる。

【００７７】

【発明の効果】以上説明した通り本発明によれば、簡潔な構成を用いて量子化／逆量子化を可能とすることができる。また、Ｎフレーム（Ｎは１以上の整数）に１回だけ累算値データを伝送すればよいので、本来の伝送すべき情報量を犠牲にすることなく、伝送効率を格段に高めることが可能となる。

【００７８】特に画像データの場合には、量子化のための除数を表わすデータをほとんど伝送しなくて済むので、その分を画質向上に振り向けることができる。

【００７９】さらに加えて、１ブロック毎に圧縮率を制御できるので、高精細な画像を記録・伝送する場合にも、特に有効である。

【００８０】また、動画像データを高圧縮率で圧縮して伝送するような場合には、その伝送データ中に占めるブロック毎のスケールファクタの符号量の大きさが無視し得ないようになる。このような場合にも、スケールファクタの伝送を省略できるので、その分データ圧縮を小さくでき、品質の良いデータ伝送が可能になる。

【００８１】さらに追加するならば目標符号量の補正を行わない場合にはバッファメモリ占有量が減少し続けることがある。これを防ぐために無効データ（ダミーデータ）を挿入する例がある。本発明は目標符号量の補正を行っているのでこのような無効データを挿入する必要がない。

【図面の簡単な説明】

【図１】本発明を適用した一実施例の構成を示すブロック図である。

【図２】目標符号量と発生情報量を比較する符号量比較器を示す図である。

【図３】累算器を示す図である。

【図４】スケールファクタ演算処理のための回路を示す図である。

【図５】符号器に含まれる除算器を示す図である。

【図６】符号化・符号量算出処理を行う回路を示す図である。

【図７】復号化・符号量算出処理を行う回路を示す図である。

【図８】復号器に含まれる乗算器を示す図である。

【図９】補正情報量を得るための構成を示す回路図である。

【図１０】従来の符号量制御装置の一例を示す図である。

【図１１】従来の符号量制御装置の他の一例を示す図である。

【符号の説明】

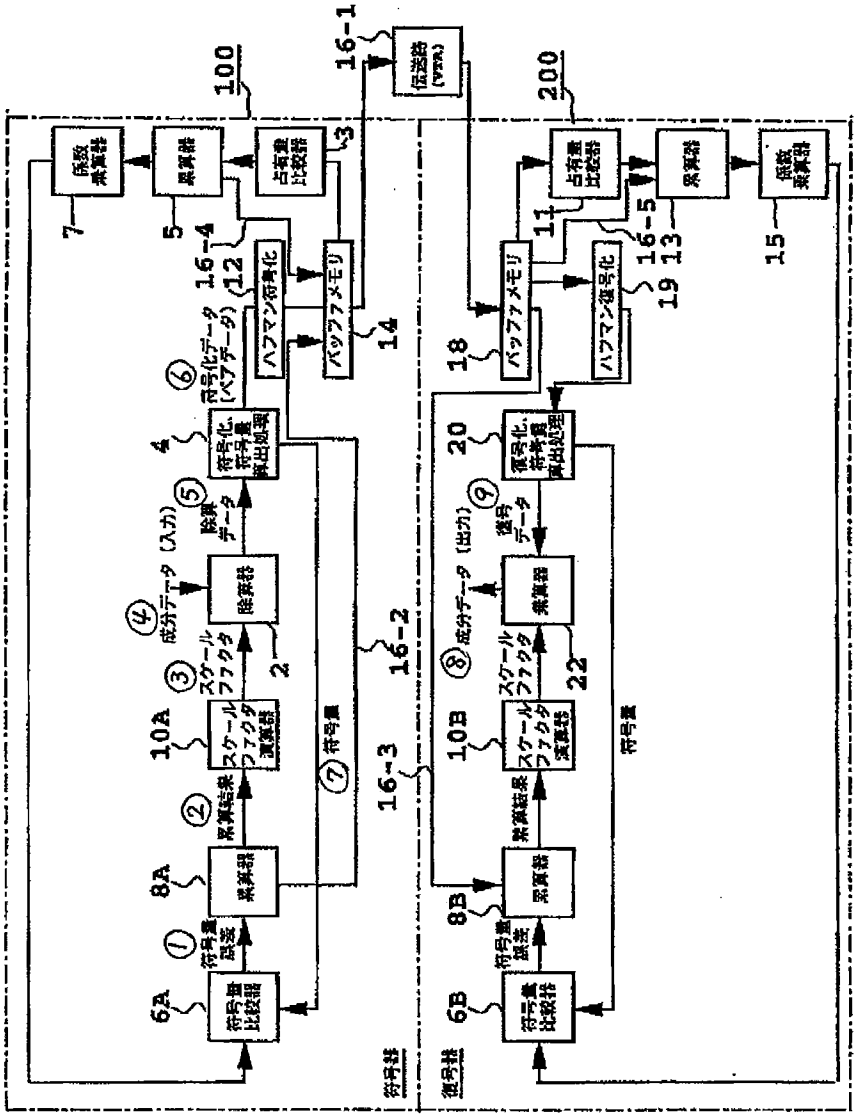
17

2　除算器
3，11　占有量比較器
4　符号化・符号量算出処理を行う回路
5，8A，8B，13　累算器
6A，6B　符号量比較器
7，15　係数乗算器
10A，10B　スケールファクタ演算器
12　ハフマン符号化回路
14　バッファメモリ

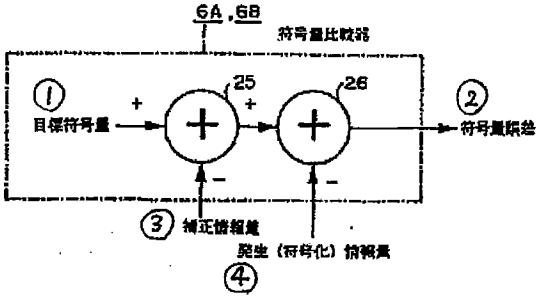
18

＊16－1，16－2，16－3，16－4，16－5　伝送路
18　バッファメモリ
19　ハフマン復号化回路
20　復号化・符号量算出処理を行う回路
22　乗算器
100　符号器
200　復号器
＊

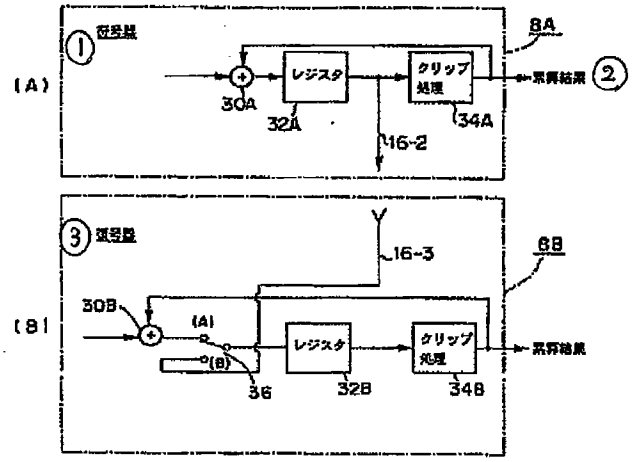
【図1】



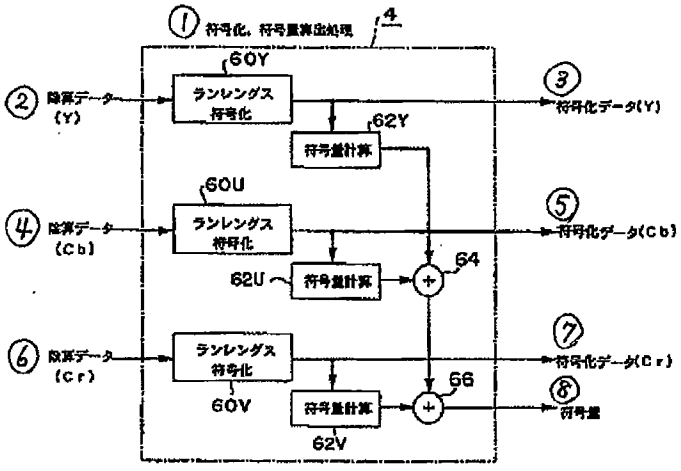
【図2】


6A,6B
符号量比較器
①
目標符号量
25
26
②
符号量誤差
③
補正情報量
発生（符号化）情報量
④

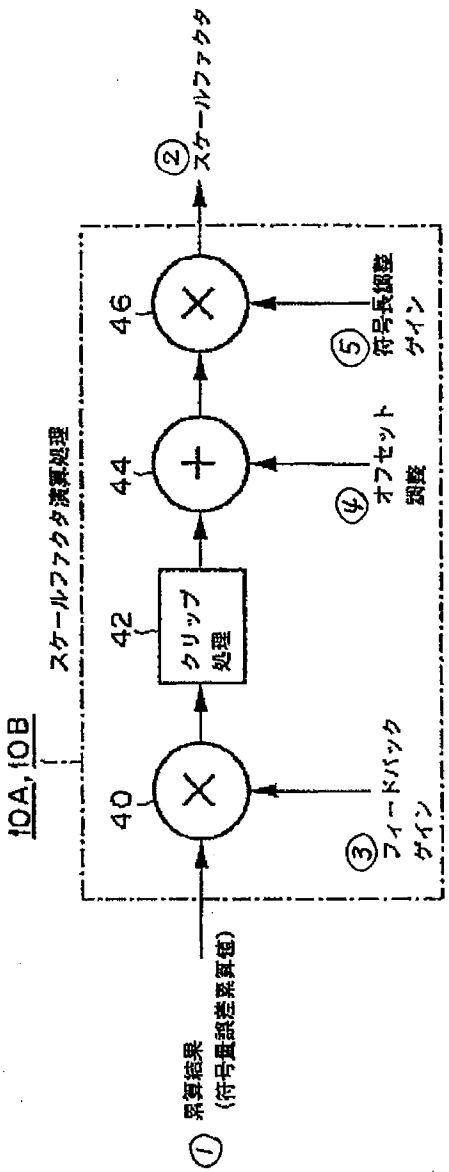
【図3】


(A)
①
8A
レジスタ
クリップ処理
累算結果
②
30A
32A
34A
16-2
③
8B
16-3
(B)
30B
(A)
(B)
レジスタ
クリップ処理
累算結果
36
32B
34B

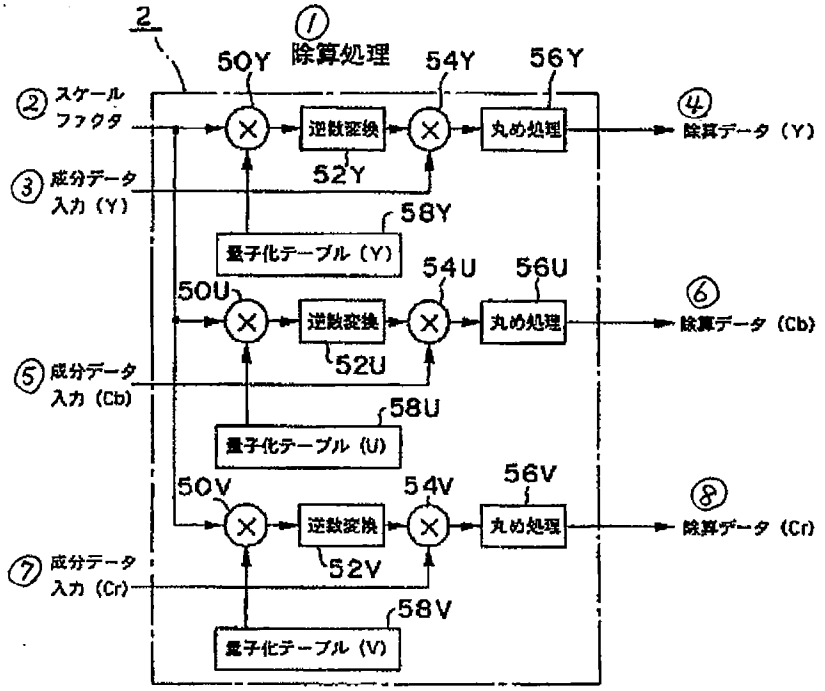
【図6】


①
符号化、符号量算出処理
4
60Y
②
(Y)
ランレングス符号化
③
符号化データ(Y)
62Y
符号量計算
60U
④
(Cb)
ランレングス符号化
⑤
符号化データ(Cb)
62U
符号量計算
64
⑥
(Cr)
ランレングス符号化
⑦
符号化データ(Cr)
60V
符号量計算
66
⑧
符号量
62V

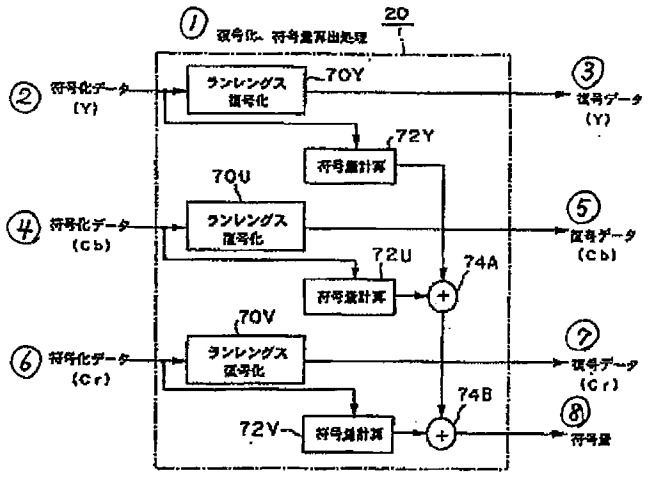
【図4】


①累算結果
(符号最話差累算値)
③フィードバック
ゲイン
④オフセット
調整
⑤符号長調整
ゲイン
②スケールファクタ
40
42
クリップ
処理
44
46
スケールファクタ演算処理
10A, 10B

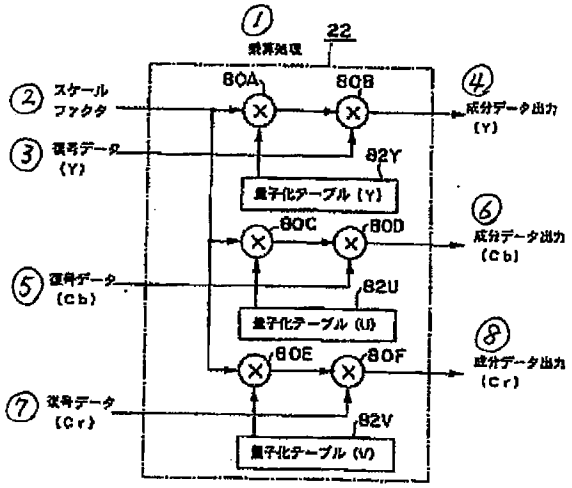
【図5】


2
①
除算処理
50Y
54Y
56Y
②スケールファクタ
逆数変換
丸め処理
④除算データ（Y）
52Y
③成分データ入力（Y）
58Y
量子化テーブル（Y）
54U
56U
50U
⑥除算データ（Cb）
逆数変換
丸め処理
52U
⑤成分データ入力（Cb）
58U
量子化テーブル（U）
50V
54V
56V
⑧除算データ（Cr）
逆数変換
丸め処理
52V
⑦成分データ入力（Cr）
58V
量子化テーブル（V）

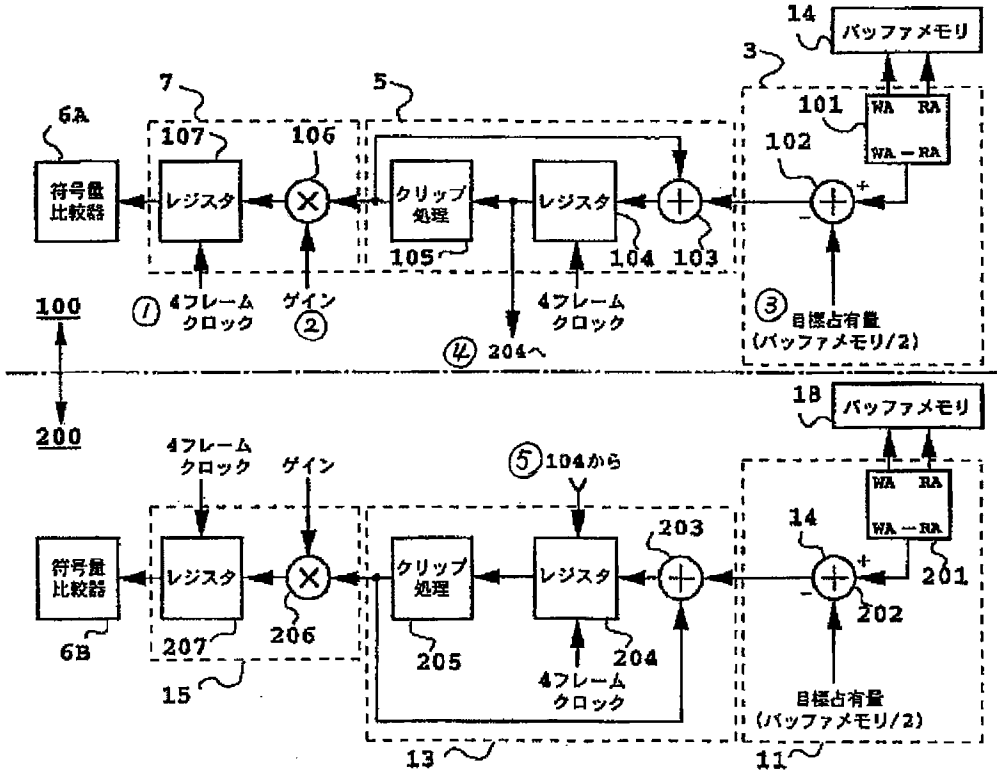
【図7】


①
20
②符号化データ（Y）
ランレングス復号化
70Y
③復号データ（Y）
72Y
符号量計算
70U
④符号化データ（Cb）
ランレングス復号化
⑤復号データ（Cb）
72U
74A
符号量計算
70V
⑥符号化データ（Cr）
ランレングス復号化
⑦復号データ（Cr）
74B
72V
符号量計算
⑧符号量

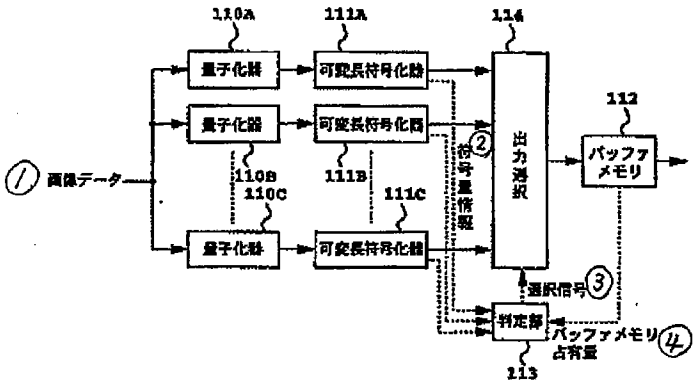
【図8】


①
22
②スケールファクタ
③復号データ(Y)
80A
80B
82Y
量子化テーブル(Y)
④成分データ出力(Y)
⑤復号データ(Cb)
80C
80D
82U
量子化テーブル(U)
⑥成分データ出力(Cb)
⑦復号データ(Cr)
80E
80F
82V
量子化テーブル(V)
⑧成分データ出力(Cr)

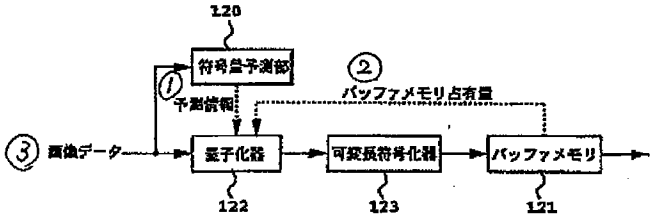
【図9】


14
バッファメモリ
3
101
WA
RA
WA－RA
102
7
5
6A
107
106
符号量比較器
レジスタ
クリップ処理
レジスタ
105
104
103
①4フレームクロック
ゲイン②
④204へ
4フレームクロック
③目標占有量(バッファメモリ/2)
100
200
18
バッファメモリ
4フレームクロック
ゲイン
⑤104から
14
203
201
202
符号量比較器
6B
207
206
205
204
15
13
11
目標占有量(バッファメモリ/2)

【図１０】


110A
111A
116
量子化器
可変長符号化器
量子化器
可変長符号化器
出力選択
112
バッファメモリ
①
画像データ
110B
110C
111B
111C
②
符号量情報
量子化器
可変長符号化器
選択信号
③
判定部
バッファメモリ占有量
④
113

【図１１】


120
符号量予測部
①
②
予測情報
バッファメモリ占有量
③
画像データ
量子化器
可変長符号化器
バッファメモリ
122
123
121